IN THE CLAIMS:

Please cancel claim 11 and amend the remaining claims as follows:

1. (Currently Amended) A method, implemented in a data processing system, for determining run-time dependencies between logical components of a data processing environment, the method comprising the steps of:

monitoring run-time activity of each of a first logical component and a second logical component of the data processing environment, said monitoring comprising determining a first activity period for said first logical component and a second activity period for said second logical component, wherein said determining comprises determining a first start time and a first end time for said first logical component and a second start time and a second end time for said second logical component;

comparing the monitored run-time activity of the first logical component with the monitored run-time activity of the second logical component to identify correlations between the monitored run-time activity of the first and second logical components, said comparing comprising comparing said first activity period and said second activity period to identify correlations between said first logical component and said second logical component, wherein said comparing comprises determining whether said first start time is before said second start time and whether said first end time is after said second end time; and

in response to identification of a positive correlation between the monitored run-time activity of the first and second logical components, recording the existence of a dependency relationship between the first and second logical components[[.]],

wherein the step of monitoring the run-time activity of said first logical component and said second logical component comprises generating events in response to completion of the processing of requests by each of the first logical component and said second logical components, and wherein the step of comparing the monitored run-time activity comprises:

calculating an activity period for each of said first logical component and said second logical components in response to generated events indicating the completion of processing of a request by the respective one of said first logical components and said second logical components; and

determining whether the activity period of said first logical component contains the activity period of said second logical component.

2. (Previously Presented) The method according to claim 1, all the limitations of which are incorporated herein by reference, further comprising the steps of:

monitoring run-time activity data of each of a plurality of logical components additional to said first and second logical components;

comparing the monitored run-time activity of the first logical component with the monitored run-time activity of the plurality of logical components to identify positive

correlations between the monitored run-time activity of the first logical component and the monitored run-time activity of any of the plurality of logical components; and

recording the existence of a dependency relationship between the first logical component and any of the plurality of logical components for which a positive correlation is identified.

3. (Previously Presented) The method according to claim 2, all the limitations of which are incorporated herein by reference, further comprising aggregating the recorded dependency relationships of the first logical component.

4. (Previously Presented) The method according to claim 2, all the limitations of which are incorporated herein by reference, further comprising:

comparing the monitored run-time activity of the second logical component with the run-time activity of each of the plurality of logical components to identify positive correlations between the monitored run-time activity of the second logical component and the monitored run-time activity of any of the plurality of logical components; and

recording the existence of a dependency relationship between the second logical component and any of the plurality of logical components for which a positive correlation is identified.

5. (Cancelled).

6. (Previously Presented) The method according to claim 1, all the limitations of which are incorporated herein by reference, wherein the comparing step comprises determining whether the activity period of the first logical component contains the activity period of the second logical component.

7. (Previously Presented) The method according to claim 1, all the limitations of which are incorporated herein by reference, wherein the step of monitoring run-time activity comprises monitoring invocations of the first and second logical components, and the comparing step comprises comparing the number of invocations of each of the first and second logical components within a monitoring period.

8. (Previously Presented) The method according to claim 1, all the limitations of which are incorporated herein by reference, further comprising:

monitoring run-time activity for a plurality of executions of each of a first and a second logical component,

comparing the monitored run-time activity for the plurality of executions of the second logical component with the monitored run-time activity for the plurality of executions of the first logical component;

determining a proportion of executions of the first logical component for which a positive correlation is identified between the compared run-time activity of the first and second logical components; and

recording in association with the recorded dependency relationship a value representing the determined proportion of executions of the first logical component for which a positive correlation is identified.

9. (Previously Presented) The method according to claim 1, all the limitations of which are incorporated herein by reference, further comprising:

monitoring run-time activity for a plurality of executions of each of a first and a second logical component,

comparing the monitored run-time activity for the plurality of executions of the second logical component with the monitored run-time activity for the plurality of executions of the first logical component;

determining a proportion of executions of the second logical component for which a positive correlation is identified between the compared run-time activity of the first and second logical components; and

recording in association with the recorded dependency relationship a value representing the determined proportion of executions of the second logical component for which a positive correlation is identified.

10. (Previously Presented) The method according to claim 8, all the limitations of which are incorporated herein by reference, further comprising:

determining a proportion of executions of the second logical component for which a positive correlation is identified between the compared run-time activity of the first and second logical components; and

recording in association with the recorded dependency relationship a value representing the determined proportion of executions of the second logical component for which a positive correlation is identified; and

generating a weight value comprising a combination function of the determined proportions of executions of the first and second logical components, and storing the weight value in association with the recorded dependency relationship.

11. (Canceled).

12. (Currently Amended) The method according to claim ~~11~~ 1, all the limitations of which are incorporated herein by reference, comprising:

computing a probability value associated with a dependency relationship by determining, for a plurality of executions of the first and second components, the proportion of executions of the first component for which the activity period of the first component contains the activity period of the second component, and recording the probability value in association with the recorded dependency relationship.

13. (Currently Amended) The method according to claim ~~11~~ 1, all the limitations of which are incorporated herein by reference, further comprising:

computing a probability associated with a dependency relationship by determining, for a plurality of executions of the first and second components, the proportion of executions of the second component for which the activity period of the first component contains the activity period of the second component, and recording the probability value in association with the recorded dependency relationship.

14. (Currently Amended) The method according to claim ~~11~~ 1, all the limitations of which are incorporated herein by reference, wherein the events are generated in response to a monitoring agent polling the first and second components for information relating to the processing of requests.

15. (Currently Amended) The method according to claim ~~11~~ 1, all the limitations of which are incorporated herein by reference, wherein the events are generated by a publish/subscribe broker which receives run-time activity data from the first and second logical components, identifies a subscriber by reference to stored subscription information, and sends the generated events to the identified subscriber.

16. (Previously Presented) The method according to claim 1, all the limitations of which are incorporated herein by reference, for monitoring a data processing system which comprises a monitoring interface for accessing run-time activity data within the data processing system, wherein the step of monitoring run-time activity comprises a monitoring agent accessing run-time activity data via the monitoring interface.

17. (Previously Presented) The method according to claim 16, all the limitations of which are incorporated herein by reference, wherein the monitoring agent comprises an aggregator for aggregating run-time activity data accessed via the monitoring interface.

18. (Previously Presented) The method according to claim 16, all the limitations of which are incorporated herein by reference, wherein the monitoring agent comprises program code for computing run-time activity metrics from the run–time activity data accessed via the monitoring interface.

19. (Previously Presented) The method according to claim 16, all the limitations of which are incorporated herein by reference, wherein the monitoring agent is configured to poll the first and second logical components via the monitoring interface.

20. (Currently Amended) A method of fault management comprising the steps of:

monitoring run-time activity of each of a first logical component and a plurality of additional logical components of a data processing environment, said monitoring comprising determining a first activity period for said first logical component and additional activity periods for said additional logical components, wherein said determining comprises determining a first start time and a first end time for said first logical component and additional start times and additional end times for said additional logical components;

comparing the monitored run-time activity of the first logical component with the monitored run-time activity of each of the plurality of logical components, to identify positive correlations between the monitored run-time activity of the first logical component and the monitored run-time activity of any of the plurality of logical components, said comparing comprising comparing said first activity period and said additional activity periods to identify correlations between said first logical component and any of said additional logical components, wherein said comparing comprises determining whether said first start time is before said additional start times and whether said first end time is after said additional end times;

recording the existence of a dependency relationship between the first logical component and any of the plurality of logical components for which a positive correlation is identified;

aggregating the recorded dependency relationships to determine a set of logical components having dependency relationships with the first logical component; and

responding to identification of a problem affecting the first logical component by analyzing the set of logical components having dependency relationships with the first logical component[[.]],

wherein the step of monitoring the run-time activity of said first logical component and said additional logical component comprises generating events in response to completion of the processing of requests by each of said first logical component and said additional logical components, and wherein the step of comparing the monitored run-time activity comprises:

calculating an activity period for each of said first logical component and said additional logical components in response to generated events indicating the completion of processing of a request by the respective one of said first logical component and said additional logical components; and

determining whether the activity period of said first logical component contains the activity period of said additional logical component.

21. (Previously Presented) The method according to claim 20, all the limitations of which are incorporated herein by reference, wherein the step of aggregating the recorded dependency relationships comprises sorting the dependencies into an order determined by a sorting heuristic, and the step of analyzing the set of logical components comprises analyzing components of the set of components in said determined order.

22. (Currently Amended) A method for determining a set of logical components of a data processing environment which are likely to be affected by termination of a first logical component of the data processing environment, the method comprising:

monitoring run-time activity of each of a first logical component and a plurality of additional logical components of a data processing environment, said monitoring comprising determining a first activity period for said first logical component and additional activity periods for said additional logical components, wherein said determining comprises determining a first start time and a first end time for said first

logical component and additional start times and additional end times for said additional logical components;

comparing the monitored run-time activity of the first logical component with the monitored run-time activity of each of the plurality of logical components, to identify positive correlations between the monitored run-time activity of the first logical component and the monitored run-time activity of any of the plurality of logical components, said comparing comprising comparing said first activity period and said additional activity periods to identify correlations between said first logical component and any of said additional logical components, wherein said comparing comprises determining whether said first start time is before said additional start times and whether said first end time is after said additional end times; and

recording the existence of a dependency relationship between the first logical component and any of the plurality of logical components for which a positive correlation is identified; and

aggregating the recorded dependency relationships to determine a set of logical components having dependency relationships with the first logical component[[.]],

wherein the step of monitoring the run-time activity of said first logical component and said additional logical component comprises generating events in response to completion of the processing of requests by each of said first logical component and said additional logical components, and wherein the step of comparing the monitored run-time activity comprises:

calculating an activity period for each of said first logical component and said additional logical components in response to generated events indicating the completion of processing of a request by the respective one of said first logical component and said additional logical components; and

determining whether the activity period of said first logical component contains the activity period of said additional logical component.

23. (Currently Amended) A computer program product comprising program code recorded on a recording medium, for controlling the operation of a data processing system on which the program code executes to determine run-time dependencies between logical components of a data processing environment, the program code comprising:

at least one monitoring agent for monitoring run-time activity data of logical components of the data processing environment, said monitoring comprising determining activity periods for said logical components, and for sending the monitored run-time activity data to a correlation identifier;

a correlation identifier for receiving, from the at least one monitor, monitored run-time activity data of each of a first logical component and a second logical component of a data processing environment, wherein said receiving comprises receiving a first start time and a first end time for said first logical component and a second start time and a second end time for said second logical component, and for comparing the monitored run-time activity data of the first logical component with the monitored run-time activity data of the second logical component to identify positive correlations between the

monitored run-time activity of the first and second logical components, said comparing comprising comparing said activity periods of said logical components to identify correlations between said first logical component and said second logical component, wherein said comparing comprises determining whether said first start time is before said second start time and whether said first end time is after said second end time; and

a dependency generator for responding to identification of a positive correlation between the monitored run-time activity of the first and second logical components by recording the existence of a dependency relationship between the first and second logical components[[.]],

wherein said monitoring run-time activity data of the logical components comprises generating events in response to completion of the processing of requests by each of the logical components, and wherein said comparing the monitored run-time activity comprises:

calculating an activity period for each of the logical components in response to generated events indicating the completion of processing of a request by the respective one of the logical components; and

determining whether the activity period of a first logical component contains the activity period of a second logical component.

24. (Currently Amended) A computer program product comprising program code recorded on a recording medium, for controlling the operation of a data processing

system on which the program code executes to determine run-time dependencies between logical components of a data processing environment, the program code comprising:

a correlation identifier:

for receiving, from at least one monitoring agent, monitored run-time activity data of each of a first logical component and a second logical component of a data processing environment, said receiving comprising receiving a first activity period for said first logical component and a second activity period for said second logical component, wherein said receiving comprises receiving a first start time and a first end time for said first logical component and a second start time and a second end time for said second logical component, and

for comparing the monitored run-time activity data of the first logical component with the monitored run-time activity data of the second logical component to identify positive correlations between the monitored run-time activity of the first and second logical components, said comparing comprising comparing said first activity period and said second activity period to identify correlations between said first logical component and said second logical component, wherein said comparing comprises determining whether said first start time is before said second start time and whether said first end time is after said second end time; and

a dependency generator for responding to identification of a positive correlation between the monitored run-time activity of the first and second logical components by generating a representation of the existence of a dependency relationship between the first and second logical components[[.]].

wherein monitoring the run-time activity of said first logical component and said second logical component performed by said monitoring agent comprises generating events in response to completion of the processing of requests by each of said first logical component and said second logical components, and wherein the step of comparing the monitored run-time activity comprises:

calculating an activity period for each of said first logical component and said second logical components in response to generated events indicating the completion of processing of a request by the respective one of said first logical component and said second logical components; and

determining whether the activity period of said first logical component contains the activity period of the second component.

25. (Currently Amended) A data processing apparatus comprising:

a data processing unit;

a data storage unit;

a correlation identifier

for receiving, from at least one monitoring agent, monitored run-time activity data of each of a first logical component and a second logical component of a data processing environment, said receiving comprising receiving a first activity period for said first logical component and a second activity period for said second logical component, wherein said receiving comprises receiving a first start time and a first end

time for said first logical component and a second start time and a second end time for said second logical component, and

for comparing the monitored run-time activity data of the first logical component with the monitored run-time activity data of the second logical component to identify positive correlations between the monitored run-time activity of the first and second logical components, said comparing comprising comparing said first activity period and said second activity period to identify correlations between said first logical component and said second logical component, wherein said comparing comprises determining whether said first start time is before said second start time and whether said first end time is after said second end time; and

a dependency generator for responding to identification of a positive correlation between the monitored run-time activity of the first and second logical components by recording in the data storage unit the existence of a dependency relationship between the first and second logical components[[.]]_.

wherein monitoring the run-time activity of said first logical component and said second logical component performed by said monitoring agent comprises generating events in response to completion of the processing of requests by each of said first logical component and said second logical components, and wherein the step of comparing the monitored run-time activity comprises:

calculating an activity period for each of said first logical component and said second logical components in response to generated events indicating the completion of

processing of a request by the respective one of said first logical component and said second logical components; and

determining whether the activity period of said first logical component contains the activity period of the second component.

26. (Previously Presented) The data processing apparatus according to claim 25, all the limitations of which are incorporated herein by reference, wherein the dependency generator is configured to generate a CIM_dependency class instance to represent an identified positive correlation, the apparatus further comprising a CIM object manager configured to record dependency information within a CIM repository in the data storage unit.

27. (Currently Amended) A distributed data processing system comprising:

a first data processing apparatus comprising a set of logical components to be monitored, and at least one monitoring agent for monitoring run-time activity data for the set of logical components and for sending the monitored run-time activity data to a correlation identifier on a second data processing apparatus, said monitoring comprising determining activity periods for said logical components; and

a second data processing apparatus comprising:

a data processing unit;

a data storage unit;

a correlation identifier

for receiving, from the at least one monitoring agent, monitored run-time activity data of each of a first logical component and a second logical component of a data processing environment, said receiving comprising receiving a first activity period for said first logical component and a second activity period for said second logical component, wherein said receiving comprises receiving a first start time and a first end time for said first logical component and a second start time and a second end time for said second logical component, and

for comparing the monitored run-time activity data of the first logical component with the monitored run-time activity data of the second logical component to identify positive correlations between the monitored run-time activity of the first and second logical components, said comparing comprising comparing said first activity period and said second activity period to identify correlations between said first logical component and said second logical component, wherein said comparing comprises determining whether said first start time is before said second start time and whether said first end time is after said second end time; and

a dependency generator for responding to identification of a positive correlation between the monitored run-time activity of the first and second logical components by recording in the data storage unit the existence of a dependency relationship between the first and second logical components[[.]]_.

wherein monitoring the run-time activity of said first logical component and said second logical component performed by said monitoring agent comprises generating events in response to completion of the processing of requests by each of said first logical

component and said second logical components, and wherein the step of comparing the monitored run-time activity comprises:

calculating an activity period for each of said first logical component and said second logical components in response to generated events indicating the completion of processing of a request by the respective one of said first logical component and said second logical components; and

determining whether the activity period of said first logical component contains the activity period of the second component.

28. (Previously Presented) The distributed data processing system of claim 27, all the limitations of which are incorporated herein by reference, further comprising at least one management application program for analyzing recorded dependency relationships.

29. (Previously Presented) The distributed data processing system according to claim 27, all the limitations of which are incorporated herein by reference, wherein the at least one management application program comprises a fault management application program.

30. (Currently Amended) A method for discovering dependencies between monitored components of a managed data processing system, comprising the steps of:

accessing, from the managed system, run-time activity data for the monitored components, said accessing comprising accessing activity periods for said monitored

components, wherein said accessing comprises accessing a first start time and a first end time for a first logical component of said monitored components and a second start time and a second end time for a second logical component of said monitored components;

comparing the accessed run-time activity data of the monitored components to identify positive correlations between the run-time activity of the monitored components, said comparing comprising comparing said activity periods to identify correlations between said monitored components, wherein said comparing comprises determining whether said first start time is before said second start time and whether said first end time is after said second end time; and

generating an identification of a dependency relationship between components for which a positive correlation is identified[[.]]_.

wherein the step of monitoring the run-time activity of said first logical component and said second logical component comprises generating events in response to completion of the processing of requests by each of the first and second logical components, and wherein the step of comparing the monitored run-time activity comprises:

calculating an activity period for each of said first logical component and said second logical components in response to generated events indicating the completion of processing of a request by the respective one of said first logical components and said second logical components; and

determining whether the activity period of said first logical component contains the activity period of said second logical component.

31. (Previously Presented) The method according to claim 30, all the limitations of which are incorporated herein by reference, wherein the step of accessing comprises accessing the run-time activity data via an API provided by the managed system.

32. (Previously Presented) The method according to claim 30, all the limitations of which are incorporated herein by reference, wherein the step of comparing the accessed run-time activity data comprises comparing a plurality of run-time activity metrics for each monitored component.

33. (Previously Presented) The method according to claim 32, all the limitations of which are incorporated herein by reference, further comprising:

computing a value representing the consistency of identification of a positive correlation between components for the plurality of run-time activity metrics; and

storing the computed value in association with the generated identification of a dependency relationship.

34. (Previously Presented) The method according to claim 32, all the limitations of which are incorporated herein by reference, wherein the step of comparing the accessed run-time activity data comprises comparing run-time activity data for a plurality of executions of the monitored components, and wherein the method further comprises:

computing a single value representing the consistency of identification of a positive correlation between components, for the plurality of executions of the components and the plurality of run-time activity metrics; and

storing the computed single value in association with the generated identification of a dependency relationship.

35. (Previously Presented) The method according to claim 30, all the limitations of which are incorporated herein by reference, wherein the step of comparing the accessed run-time activity data comprises comparing run-time activity data for a plurality of executions of the monitored components, and wherein the method further comprises:

computing a value representing the consistency of identification of a positive correlation between components for the plurality of executions of the components; and storing the computed value in association with the generated identification of a dependency relationship.